American Journal of Computer Architecture 2024, 11(6): 75-82
DOI: 10.5923/j.ajca.20241106.01

Design, Implementation and Importance of Personal Finance Management (PFM) for Financial Institutions

Pradeep Jain

Principal Application Architect, Discover Financial Services, Pittsburgh, PA, USA

Abstract In the modern fintech revolution, most financial institutions are working towards providing lucrative functionalities to their customers on mobile and web channels to increase their deposits, customer base, customer loyalty and business. One of the functionalities that Banks should look forward to providing is Personal Finance Management tools through which customers can manage their finances, set financial goals, set budgets, track expenses and get help achieving financial freedom. This article will shed light on designing, implementing and the importance of personal finance management for financial institutions which help banking customers to achieve financial freedom and insights into their financial health.

Keywords PFM, Insights, Push, Pull, Hybrid, Financial

1. Introduction

Keeping personal financial management in applications portfolio, banks can provide their customers with a comprehensive view of customers finances across different financial institutions, customer's spending patterns and behaviour, alerting them about the credit and debit transactions, proposing ideas of saving expenses, and providing options of setting budgets and goals. Through all intuitive functions, banks can gain their customer loyalty, increase the number of finances and therefore increase overall revenue leading to other major investments. Customers get information about their spending habits and leading to being aware of the unnecessary expenses which customers might be overlooking. For example, keeping subscriptions of similar services might not be in the consumer's favour. This article will explain the software design of Personal Finance Management along with implementation details and the importance of the product through statistical data from different financial institutions. across the US and Europe.

2. Importance of PFM for Financial Institutions

The importance of PFM for financial institution has been recognized due to several factors as below

* Corresponding author:
pradeep3j@gmail.com (Pradeep Jain)
Received: Nov. 19, 2024; Accepted: Dec. 3, 2024; Published: Dec. 11, 2024
Published online at http://journal.sapub.org/xxx

2.1. Enhanced Customer Experience

The customer experience gets enhanced after using Personal Finance Management tools as it provides comprehensive view of customer financial assets with in and across financial institutions. It self-empowers them to take control of their finance and get personalized advices based on their spending patterns. The ability to set budgets and make financial goals make customers more diligent towards meeting them by doing savings. The possibility of using aggregators and bringing customer financial data from other financial institutions help them to make decisions towards meeting their goals and avoid making grave financial mistakes. The transparency gets achieved as alerting them about duplicate expenses after looking their expenses builds trust between financial institution and customer.

2.2. Customer Retention, Originations Growth

Bringing PFM tools in application suite assists banks in uplifting customer loyalty as it embeds financial institution in customer's daily life by providing them options of setting goals and budgets which can narrow down to weeks or months or can be long term of years. The PFM inclusion differentiates financial institution in a competitive market especially among tech savvy customers who rely on modern mobile apps in receiving financial insights.

2.3. Revenue Growth

PFM generates insights across financial products owned by customer like personal loans, student loans, credit cards etc. Therefore, by providing relevant insights to customer about spending patterns and providing personalized advice, customers are more likely to engage themselves in other

financial banking products like mortgage, investment accounts **etc.** which brings more revenue to financial institution. The customers who use personal finance management products are more likely to be engaged and hence banks can generate more revenue by offering other banking products and managing their complete financial portfolio.

2.4. Regulatory and Social Responsibility

It's a social responsibility of financial institutions to educate customer about finance management and PFM comes handy with that. Offering PFM tools also align with regulatory emphasis of promoting financial literacy.

3. PFM Market Size and Projections

The Personal Finance Management Software Market Size was valued as $2.68 billion in 2023 and is expected to rise to $7.80 Billion by 2032 at CAGR ratio of 12.5% during the forecast period. The market for personal money management software is growing quickly due to rise in financial education and need of financial goals and budgeting. The smartphone usage due to internet has grown substantially which enhance access to technological products that provides really time tracking of expenses and providing financial advices. People look for creative ways to handle their finances and gets allured to the 360-degree view of their financial health and the insights and recommendations generated by PFM products.

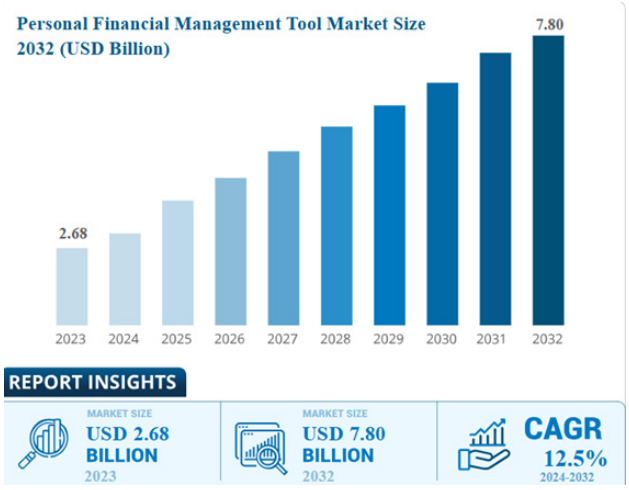

Figure 1. A sample line graph showing PFM expected growth

4. Key Components of PFM

4.1. User Interface

The UI would be the front face of PFM product on financial institution digital channels like Web, tablets and Mobile. The UI would have widgets with intuitive dashboard containing graphs, charts, images for showing financial health, spending, savings and recommendations. Most of the times, organizations can either choose to create UI on their own or can outsource to receive SDK provided by certain organizations which provides expertise in PFM products.

4.2. Insights-Generator

The insights generator would be the component within PFM suite which would process the customer transactional data over a period of time and would generate meaningful determinations from data. For Ex: it would look for customer transactions from saving, checking, and money market accounts from last 6 months and would categorize them in grocery, entertainment, transportation etc. to showcase on UI. While massaging the data, PFM would look for duplicate transactions and might alert to the customer. The insights generator is also responsible for managing customer goals and budgets by keeping track of customer preferences or earmarking the customer deposits based on their goals settings on specific accounts.

4.3. Core Banking Platform

Core banking platform would be the transactional data source of the financial institution which manage all customer deposit accounts (Saving, Checking, Money Market, IRA, CD) and transaction data that has been posted in those accounts. The core banking platform is the heart of bank which would have all banking accounts and transactions (scheduled or recurring) associated with customer profile.

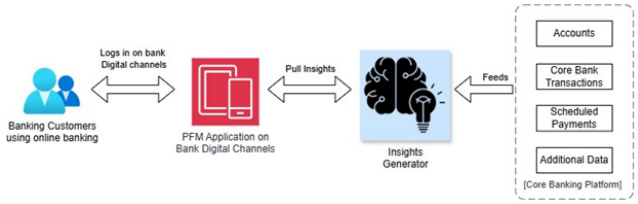

Figure 2. Image showing key components of PFM

5. PFM Design Option - PUSH

The PUSH style of design for personal finance management contains the strategy for pushing the customer data (Accounts, core banking transactions, scheduled payments and additional data) from core banking platform to Insights- Generator every night. The nightly batch process can be built in the technology of choice by the financial institution which can be executed at fixed point of time to pull the customer financial data and push to Insights-Generator component. The insights-generator component can process the data and be ready with insights, recommendations and financial advices based on customer financial health and transactions retrieved from nightly batch transmission.

The nightly batch would contain a scheduler that would kick off every night, would read all the enrolled customer ids from enrollment Micro Service followed by requesting core banking platform for its transactions. The process would create file contained all enrolled customer's accounts and its transactions to be delivered to Insights-Generator.

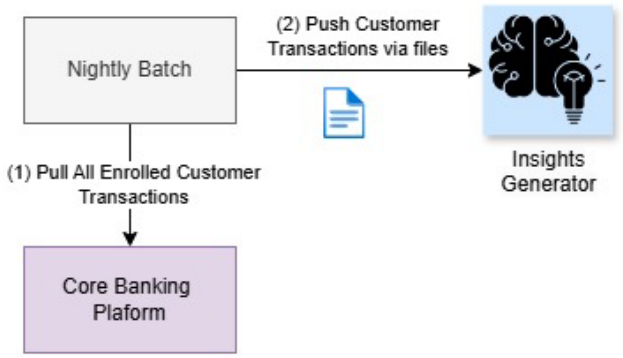

Figure 3. Image showing PUSH approach of sending customer data via files

As and when customer logs in, the UI component, which can be built on modern technologies like REACT, can make HTTP calls to PFM backend end microservices. The PFM microservices would expose various functions like verifying eligibility, enrollment etc. before they can pull the pre-ready insights from Insights-Generator component and forward them back to UI component for display to customers.

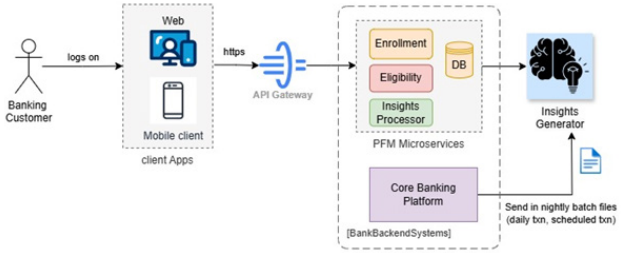

Figure 4. Image showing PUSH approach for PFM

5.1. Pros of Push Approach

5.1.1. Reduced Latency

The latency refers to the time taken starting from user/ customer taking action on user interface on a web/mobile application till the time the customer receives the desired response. Since the customer financial data is already available in Insights-Generator and has been processed beforehand, therefore, all insights would be pre-ready as soon as customer logs in to display on customer profile leading to reduced latency.

5.1.2. Product Growth

The product growth would not be concerning because if customer enrollment increases, then data in files will increase for sending out the customer data. If the secured transmission channel has been established then available file storage can be increased easily. With cloud adoption, the cost of securing data storage is relatively cheaper.

5.2. Cons of Push Approach

5.2.1. Scalability

The nightly batch would need to contain a scheduler which can be scaled only vertically because multiple schedulers on same file would end up in racing condition and files being non transactional can be processed only by one single scheduler.

5.2.2. Delayed Insights

The financial institutions would mostly need the enrollment confirmation before financial insights and recommendations can be delivered on customer digital channel. For doing so, one of the micro services would need to expose the functionality. Whenever customer enrolls in the program, the insights can be shown only the next following day because the batch will run at night and therefore, from next day onwards customer would be able to see insights and recommendations.

5.2.3. Delayed Transaction Updates

Most of the transactions would happen during day time, therefore, if there is significant credit/debit transaction on the account, then customer will not be immediately receiving insights for that. Though, the credit/debit communication would still be sent to the customer as part of fraud protection which is out of scope for discussion of personal financial management.

5.2.4. Significant Error Impact

If transmission failure occurs in nightly batch program to deliver files from core banking platform to Insights-Generator then the scope of impact would be significant because none of the enrolled customer would be able to receive most updated insights as without customer data, Insights-Generator cannot deliver the most updated recommendation and financial advices. Therefore, the impact of failure is much higher in case of PUSH strategy.

5.2.5. Security Requirement of Data at Rest

Since files are considered as data at rest, therefore, the data within file would need to be in encrypted format due to them containing confidential customer accounts data. Since The critical data elements cannot be stored in plain text and therefore payload encryption would be extra over the encrypted transmission channel, which anyway is needed.

5.2.6. Higher Transmission Load

In PUSH approach, all enrolled customer accounting and transactional data would need to be transmitted to Insights-Generator irrespective of customer's login activity. There is a possibility that customer doesn't frequently login to the online banking but PUSH approach will continue transmitting customer data every night regardless of customer's interaction with online banking.

Considering CONS over PROS, the PUSH approach is not favourable while designing Personal Finance Management Solutions. But Financial Institution's architect can decide on PUSH approach as well considering the possibility of reduced latency with compromised customer experience and higher risk of impact in case of failure. The architect can decide to increase the scheduler frequency like hourly to pull

and send out transaction batch to make customer experience relatively better but that would also be at cost of pulling data during business hours and that too for all enrolled customers which might be an overkill to core banking platform.

6. PFM Design Option - PULL

The PULL design style includes the Insights-Generator to be pulling the customer accounts and transaction history only when the customer logs in the online banking. As and when customer access online banking, the UI component would need to verify the enrollment via PFM Micro service post which the UI component, can send request to Insights-Generator for generating the insights. The amount of data i.e. X month of data, which should be pulled and considered for generating the insights would need to be decided by the business.

The insights generator would make HTTP request to core banking API to request customer accounts and transaction history, would process the data and would generate insights to be sent back to UI. The architect should choose to request only the delta load i.e. request transactions only after the time customer has last logged in and historical transactions data should be cached in Insights-Generator component.

The Insights-Generator products delivered by market leaders commit to have SLA of 200-900 millisecond to deliver insights after they receive customer transactions, hence doing the complete process in one single HTTP call i.e. to receive request from UI to Insights-Generator followed by its request to Core Banking Platform to receive the data and then to generate and deliver insights back to UI might not be as concerning as it might sound.

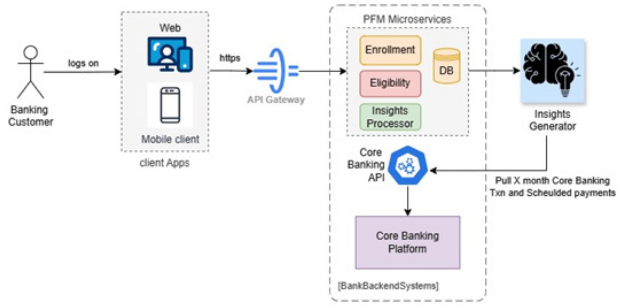

Figure 5. Image showing PULL approach for PFM

6.1. Pros of Pull Approach

6.1.1. Scalability

The PULL style makes the complete system capable of being horizontal scalable as the core banking API will be exposed using HTTP protocol and therefore, its instances can be increased by deploying them using modern deployment strategies in dockers and using Kubernetes. The PFM micro services can be scaled out as well as needed based on the realized load. The Insights-Generator and Core Banking Platform are built inherently to process higher workloads.

6.1.2. Instant Insights and Enhanced Customer Experience

In PULL architectural style, the insights would be immediately generated as soon as transaction is done on customer account and online banking is accessed by the customer. If customer makes a significant grocery transaction which surpass the grocery budget set by the customer, then the insight will immediately get generated for alerting customer about grocery budget over expense. Similarly, if customer has set a goal and a significant credit meets the goal then customer will immediately get insights after accessing online banking. Therefore, having PULL approach enhances the customer experience.

6.1.3. Resource and Cost Preservation

The resources like memory, CPU, network etc. would be used as needed in PULL architectural style which would save COST to the financial institution. Unlike PUSH, the transaction data would be pulled only for customers who are accessing online banking instead of pulling for all customers which leads to providing data on demand only. This would save unnecessary resource utilization and therefore assist in resource preservation and cost saving.

6.1.4. Reusability and Time to Market

In any Financial Institution providing digital banking, there would be existing API which would request transaction data (core banking or scheduled payments) to support the banking home page like showing transaction history, available balance etc. To support PULL style, the existing API should suffice and, in most cases, doesn't need major rehaul of services. Therefore, banks would save development time and would have faster time to market due to reuse of existing components.

6.1.5. Maintainability

The PULL style is relatively easier to maintain because it contains the microservices and existing core banking API's which are simple to maintain especially after modern deployment strategies of dockers and Kubernetes instead of dedicated schedulers and file management solutions which were required in PUSH.

6.1.6. Reliability

The reliability would be higher in PULL style due to reuse of existing stable API and much lesser point of failures. In PUSH, there are extra point of failures like schedulers and file management software.

6.1.7. Interoperability

The PULL style has only the API through which data needs to be transmitted to Insights generator but PUSH style would need to have schedulers and file management software which

might have higher interoperability issues.

6.2. Cons of Pull Approach

6.2.1. Latency

In most cases, financial institutions choose for insights generator to be outsourced to organizations who are excel in creating AI (Artificial Intelligence) and ML (Machine Learning) software. This means that insights generator component might be deployed on SaaS i.e. on cloud infrastructure. This is crucial to understand that any system which is outside of Bank periphery would need to be first onboarded for security implications before any customer data is released for insights generation. This means overhead of authentication and authorization of system which leads to higher latency. The proven architecture for authentication and authorization is federated model, which means that security token would be issued to Insights-Generator after it validates itself via credentials provided during onboarding to 3rd party security system. Doing the security check means that customer will be waiting while its data can be released to Insights-Generator to get processed for insights generation leading to higher latency and therefore leads to customer dissatisfaction.

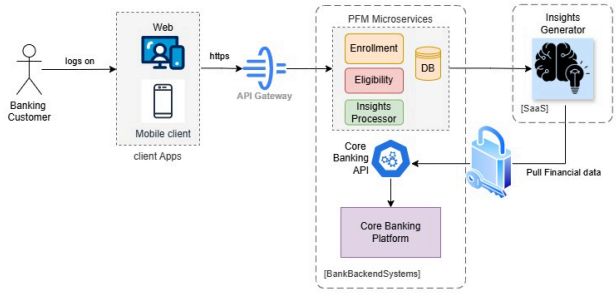

Figure 6. Image showing higher latency due to strict security for SaaS systems

6.2.2. Outlier Customer Handling

There might be customers in the financial institution who have higher number of accounts. The first API request that insights generator would need to make would be of account numbers followed by having transactions for each of the listed accounts. Due to this requirement, there is possibility of much higher latency for the customer who are outliers. Therefore, a careful analysis needs to be performed about the percentage of customer holding the higher number of accounts so that impact assessment can be done. The solution that can be adopted is to use parallel API request for transaction data and use caching but making a sequential accounts request cannot be avoided.

Number of Accounts	% of customers
1-5	75%
5-10	15%
10-20	5%

Figure 7. Image showing analysis to be done for percentage of customers having higher number of accounts

7. PFM Design Option - HYBRID

The HYBRID architectural style aims to achieve personal finance management with best of both worlds. The idea is to get the critical data to show the immediate insights via PULL approach and get the historical weekly data via PUSH approach. This means that scheduler for PUSH would need to run once in a week to fetch the weekly transactions and send over to Insights-Generator and as and when customer access online banking then 7 days' worth of data should be pulled over real time. The scheduler time can be set during non-business days and hours which would not put heavy lift workload on the system. Though there is no free ride and hybrid approach will also have pros and cons.

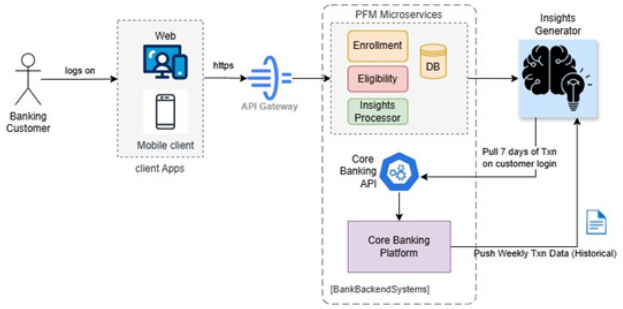

Figure 8. Image showing Hybrid approach for PFM

7.1. Pros of Hybrid Approach

7.1.1. Reduced Latency

The latency concern as reflected in PULL approach will not be as bothersome because hybrid design approach is pulling only 7 days of transaction data instead of months of data required for generating spending patterns and habits of the customer to provide effective financial advices. The rest of the data is getting pushed weekly via files and has no impact on latency.

7.1.2. Real Time

As Hybrid still relies on PULL approach for fetching last 7 days of data therefore, the system will be real time as well maintaining convenient customer experience.

7.1.3. Outlier Customers

The customers carrying higher number of transactions will not be an issue because of lightweight pull of 7 days of data. The first API request of pulling number of accounts would still produce some latency but not impactful as in PULL approach because of significant reduction in pulling transactional data.

7.2. Cons of Hybrid Approach

7.2.1. Expensive

The hybrid approach is expensive to develop and maintain. The reason being that all components required to support PUSH and PULL approach will need to be developed, deployed and maintained. The hybrid approach becomes

overly expensive because the customer transaction data needs to be fetched from 2 different channels, one being batch and another one being API exposed over core banking platform. The outsourced insights-generator subscription fee is higher if it's contracted to fetch data from multiple channels due to overheads of deduplication and multiple point of referencing financial data. The Hybrid approach needs API on core banking platforms equivalently to PULL approach, which means the complete development, deployment, maintenance cost is incurred even though the usage is diminished due to pull of only 7 days' worth of data. Moreover, since the insights-generator is outsourced component, hence all security implications will need to be considered before sharing customers financial data. The hybrid approach maintenance cost goes higher due to issue of file transmissions because one single file transmission failure means that none of the customers data will be shared with insights-generator leading to diminished customer experience. Therefore, financial institutions will need to subscribe for higher availability systems due to larger impact leading to higher cost.

7.2.2. API Scalability and Transaction Management

The hybrid approach needs best of both worlds which means that it requires development and maintenance of API on core banking platforms to provide weekly customer data. The financial institution would still need to invest on the scalability infrastructure as API demand would variate based on the customer weekly transaction volume. If API are deployed on-prem then upfront cost would need to be incurred for scalability needs or otherwise financial institution would need to contract with cloud provider for on demand API usage.

8. Key Considerations

8.1. Pagination Size

There is possibility of API timeouts for the customers if the number of transactions is too high. Therefore, pagination can be planned to pull only limited number of transactions in one request. The magic number of transactions to be pulled in each request should be determined based on the production data analytics. The idea should be to cover majority of customers to receive transaction data in 1st API request. For Ex: If 98% of customers are making 350 transactions in the time (Ex: 3 month) required for generating insights then that can be the magic number. If architect ends up keeping lower limit like 100 then there would be 4 API requests needed to fetch 350 transactions bringing latency to the system.

8.2. Inhouse vs Outsource (Insights-Generator)

Financial institutions need to consider is if they want to build Insights Generator component or want to outsource it to 3rd party vendors offering PFM products. It's not one size fits all situation and the decision depends on long term and short-term goals of each organization. In most of the cases, the development of Insights generator needs one-time heavy investment due to developing AI and ML capabilities in the product to provide AI powered insights and reward/risk ratio might need to be evaluated before making the decision.

8.3. SaaS vs OnPrem (Insights-Generator)

If financial institutions choose for outsourced product of Insights-Generator then the decision would need to made to install the product On-Prem or on SaaS environment. Organization needs to consider the upfront infrastructure like server configurations, database management, data security before making decision on On-Prem installations. The SaaS offering is favourable if financial institutions wish to evaluate the product on production before making heavy lift investment or might have consideration to move to different vendor later on.

9. PFM Implementation

9.1. Eligibility Check

The first stage of implementation is to verify customer eligibility for personal finance management. In most of the cases, financial institutions need minimum criteria for customers to have either of Saving, Checking or Money Market accounts and having positive balance in last couple of months. The banks home page UI should need to make API request for eligibility micro service to check eligibility via pulling account information and balances from core banking platform. Post eligibility verification, option of enrollment has to be provided.

9.2. Enrollment

If customer is determined to be eligible for PFM, then enrollment options can be provided. As customer enrolls, the enrollment flow should generate a consent-Id that can be maintained in Enrollment micro-service database for quick verifications and has to be sent out to Insights-Generator. The insights-generator has to send out same consent Id while pulling the customer information which should be validated at core banking platform API before rendering any customer data out.

9.3. Disclosures

As per financial regulations, banks have to present disclosure to customers if their financial data is exposed to third party vendors for Insights Generation. Therefore, if financial institution has opted for leveraging 3rd party Insights-Generator the disclosure would need to be presented and challenged to be accepted before any financial information can be shared for generating insights, setting budgets and goals, and providing financial advices. The implementation of Disclosure management is out of scope for this discussion but can be implemented via database having mapping of customer ids and disclosure versions or using cloud storage solutions like S3 or having vendor specific disclosure

management solutions.

9.4. Insights-Processor

The UI component after successful enrollment should pass control to micro service of insights processor which would reverify the consent for the customer-id and should make request to Insights-Generator with consent Id to pull customer financial data. The insights-generator would use the consent-Id while pulling information from core banking platform to be validated. Once customer financial information is received then Insights-Generator would generate insights (JSON format) and would pass on response to Insights-Processor to show insights to customer on UI.

9.5. Data Analytics and Feedback

Financial Institutions should also collect analytics for measuring success of Personal Finance management offering. For doing that, the enrollment data can be pushed over to data warehouse like snowflake to generate analytics. In addition to that, generated insights should also have option of feedback that customers can use to provide their take on insights which should also be collected in snowflake for measuring out the worthiness of the program.

10. Design Recommendation

The recommendation would be to use Hybrid approach if Financial Institutions is willing to spend extra cost of having best of both worlds of Push and Pull. Otherwise, PULL approach is favourable considering the PROS and CONS of each approach listed above.

The IT architect needs to make the cost-benefit analysis before choosing the Hybrid approach. On an average, the response times for a typical customer having 300 transactions in 6 months would be 100 milliseconds in Push Approach, 300 milliseconds in Hybrid and 2000 milliseconds in Pull Approach. Therefore, Push is fastest and Pull is slowest while Hybrid takes the middle stand but being most expensive. Per table below, it can be realized that hybrid approach will turn out to be most expensive because of higher components to be maintained and higher subscription fee due to multiple channels, Batch and API, for customer data.

Expense	Push	Pull	Hybrid
Scheduler	Y	N	Y
File Transmission	Y	N	Y
API on core banking platform	Y	Y	Y
Secuity Federated Model for SaaS onboarding	N	Y	Y
Subscription Fee	Y	Y	Y+

Figure 9. Table showing needed components for each approach leading to higher cost

11. PFM Adoption

While the Personal Finance Management product is beneficial for financial institutions through various dimensions like enhancing customer experience, higher customer retention, accounts originations growth leading to higher revenues but it also comes with certain challenges as listed below especially for relatively smaller financial institutions.

11.1. Compliance

The first challenge is to share customer's financial data with outsourced insights generator which needs to be compliant with financial privacy standards. Financial institutions need to have legal departments to look after the compliance issues which is not feasible sometimes for smaller financial institutions.

11.2. Expensive

The management and development of Personal Finance Management product is expensive as it incurs the cost of outsourced Insights-Generator subscription which needs to be paid by financial institution regardless of the benefits attained via the product. In case, the product is not heavily used by the customers or it doesn't lead to higher deposits then initial investment and the subscription fee becomes the overhead on the financial institution.

11.3. Resource Utilization

The smaller financial institutions have limited quarterly and yearly budgets for IT needs which might be earmarked for initiatives which are more prioritized than adoption of Personal Finance Management. The human and technical resources might be overbooked and therefore, keeping them engaged on PFM might not be practical for the institution.

11.4. Complexity in Current Landscape

The PFM adoption means pulling customer transaction data, organizing them in various spending categories and showcasing them in eye catching ways to customer while keeping analytics for enrollments and revenue growth due to PFM product. The implementation requires integration with current systems in Financial Institution and if Bank is using legacy systems, then it might not be feasible to pull extensive customer data as legacy systems might not be supportive for modern architectural styles.

11.5. ROI Assessment

The return of investment needs to be justified for risk-benefit analysis. It happens that sometimes risks are not assessed properly leading to making higher investments on the products which are not on priority with Bank long term goals. The cost of misjudgement of ROI is much higher for smaller institutions rather than large ones as the absorption capacity is much lower for smaller institutions.

12. Case Study

U.S. Bank has launched Personal Finance Management solution with name "Smart Assistant" which offers the full

view of customer finances in one place by using the aggregations. It monitors customer spending with money tracker across all accounts with weekly and monthly snapshots. It sorts customer transactions by category and view spending trends over time. It identifies patterns and opportunities with personalized insights for the customer.

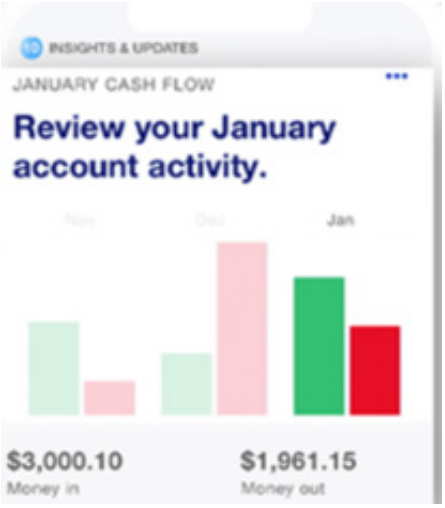

Figure 10. Sample of Insight from U.S bank

13. Conclusions

Personal Finance Management tools are needed for enhancing customer engagement, loyalty, and increasing financial literacy. It helps getting banking business and growth in revenue while at the same time assisting customers to make informed decisions about their financial health and to provide personalized assessments and advices based on the spending patterns. Even though, there are multiple approaches to make software design for PFM, being PUSH, PULL or HYBRID, the ideal choice depends on the PROS and CONS of each approach as realized by financial Institution. The careful design and implementation of PFM tools can help to enhance bank's digital transformation journey.

ACKNOWLEDGEMENTS

The heading of the Acknowledgment section and the References section must not be numbered.

SAP Productions wishes to acknowledge all the contributors for developing and maintaining this template.

DISCLOSURE

This section is ONLY for those who requested disclosure. The name of the experts that reviewed your paper, in case they accepted selling disclosure to you, will appear here. Each reviewer is allowed to make their own price for that, since that is a public endorsement of your findings and may be used for varied purposes.

REFERENCES

[1] Personal Finance Management Tools 2024. [Online]. Available: https://www.emarketer.com/content/personal-finance-management-tools-2024/.

[2] PERSONAL FINANCIAL MANAGEMENT TOOL MARKET REPORT OVERVIEW [Online]. Available: https://www.businessresearchinsights.com/market-reports/personal-financial-management-tool-market-111515.

[3] U.S Bank [Online] Available: https://www.usbank.com/online-mobile-banking/personal-finance.html.

[4] Pradeep Jain, Implementation of ACH Notification of Change, *American Journal of Computer Architecture*, Vol. 11 No. 4, 2024, pp. 48-52. doi: 10.5923/j.ajca.20241104.03.

[5] Malik Syed, Implementation of Gather Stats in ETL Informatica Workflows for Oracle Database Performance Tuning, *American Journal of Computer Architecture*, Vol. 11 No. 4, 2024, pp. 53-58. doi: 10.5923/j.ajca.20241104.04.

[6] Malik Syed, Mainframe Modernization Strategies, *American Journal of Computer Architecture*, Vol. 11 No. 1, 2024, pp. 1-9. doi: 10.5923/j.ajca.20241101.01.